4-1-2006

Wavelet Deconvolution in a Periodic Setting Using Cross-Validation

Leming Qu
Boise State University

Partha S. Routh
Boise State University

Kyungduk Ko
Boise State University

This document was originally published by IEEE in *IEEE Signal Processing Letters*. Copyright restrictions may apply. DOI: 10.1109/LSP.2005.863657

Wavelet Deconvolution in a Periodic Setting Using Cross-Validation

Leming Qu, Partha S. Routh, and Kyungduk Ko

***Abstract*—The wavelet deconvolution method WaveD using band-limited wavelets offers both theoretical and computational advantages over traditional compactly supported wavelets. The translation-invariant WaveD with a fast algorithm improves further. The twofold cross-validation method for choosing the threshold parameter and the finest resolution level in WaveD is introduced. The algorithm's performance is compared with the fixed constant tuning and the default tuning in WaveD.**

***Index Terms*—Cross-validation (CV), wavelet deconvolution.**

I. Introduction

RECENT work by Johnstone *et al.* [4] has introduced WaveD: the method of wavelet deconvolution in a periodic setting. WaveD uses band-limited wavelets that offer both theoretical and computational advantages over traditional compactly supported wavelets. The translation-invariant version of WaveD [2] improves the performance of ordinary WaveD by cycle-spinning over all circulant shifts. The fast algorithm that implements the translation-invariant version of WaveD takes full advantage of the fast Fourier transform (FFT) and runs in $O(n(log(n))^2)$ steps only. The excellent asymptotic results and fast algorithm make WaveD a very attractive noniterative deconvolution technique. We refer WaveD as the translation-invariant version of WaveD below if we do not explicitly distinguish them.

This letter introduces a twofold cross-validation (CV) method for choosing the threshold parameter and the finest resolution level in WaveD. Section II reviews the WaveD method. The CV algorithm is introduced in Section III. Section IV illustrates the algorithm using simulation. All the discussion in this letter concerns signals in one dimension but can be extended to higher dimensions naturally. See [3] for WaveD image deblurring, a two-dimensional case.

II. WaveD: Wavelet Deconvolution in White Noise

In the periodic setting and discrete data, the deconvolution can be stated as follows. Suppose we observe

$$y_i = f \star g(t_i) + \sigma n^{-\frac{1}{2}} z_i, \quad i = 1, 2, \ldots, n \tag{1}$$

Manuscript received September 3, 2005; revised November 14, 2005. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Arie Yeredor.

L. Qu and K. Ko are with the Department of Mathematics, Boise State University, Boise, ID 83725 USA (e-mail: qu@math.boisestate.edu; ko@math.boisestate.edu).

P. S. Routh is with the Department of Geosciences, Boise State University, Boise, ID 83725 USA (e-mail: routh@cgiss.boisestate.edu).

Digital Object Identifier 10.1109/LSP.2005.863657

where $t_i = i/n \in T = [0, 1]$, σ is a positive constant, z_i's are independent and identically distributed (i.i.d.) white Gaussian noise, and

$$f \star g(t) = \int_T f(t-u)g(u)du. \tag{2}$$

The goal is to recover the unknown function f from the noisy, blurred observations $y = (y_1, \ldots, y_n)^T$ given the known function g. It is further assumed that the function $f \in L^2(T)$ is periodic on the unit interval T, and g has a certain degree of smoothness quantified by the decay of its Fourier coefficients G_l. For ordinary smooth convolution, $|G_l| \sim |l|^{-\nu}$ and the decay parameter ν is referred as the Degree of Ill Posedness (DIP) of the deconvolution problem (1) according to [4] (the notation $a_k \sim b_k$ means that $lim_{k\to\infty}(a_k/b_k) = C$ for a constant C).

The ordinary WaveD estimator of f, based on hard thresholding, is

$$\tilde{f} = \sum_{\kappa\in I_0} \tilde{\alpha}_\kappa I\left\{|\tilde{\alpha}_\kappa| \geq \lambda_{j_0}\right\}\Phi_\kappa + \sum_{\kappa\in I_1} \tilde{\beta}_\kappa I\left\{|\tilde{\beta}_\kappa| \geq \lambda_j\right\}\Psi_\kappa.$$

The $I\{\cdot\}$ is the usual indicator function. The $\tilde{\alpha}_\kappa$ and $\tilde{\beta}_\kappa$ are estimated wavelet coefficients of the true wavelet coefficients α_κ and β_κ of f, respectively. $I_0 = \{(j_0, k) : k = 0, 1, \ldots, 2^{j_0} - 1\}$ is the set of indexes corresponding to a coarse resolution level j_0 and the set of indexes $I_1 = \{(j, k) : k = 0, 1, \ldots, 2^j - 1, j_0 \leq j \leq j_1\}$ details up to a fine resolution level j_1. The λ_j is the threshold for the estimated wavelet coefficients at the jth resolution level. The chosen scaling function Φ and wavelet function Ψ are band-limited. In particular, the periodic Meyer wavelet basis is used. The algorithm implementing the discrete Meyer wavelet transform is different from the pyramid algorithm of the usual discrete wavelet transform, which uses the compactly supported wavelet basis [5]. The Kolaczyck's algorithm for discrete Meyer wavelet transform operates on the Fourier coefficients of both the data and the Meyer wavelet and takes $O(n(log(n))^2)$ steps.

With a slight abuse of notation, mainly in order to be consistent with the notation in [4], denote Y_l and Ψ_l^κ as the Fourier coefficients of y and Ψ_κ, respectively. Denote G_l as the Fourier coefficients of g. Denote $C_j = \{l : \Psi_l^\kappa \neq 0\}$ and its cardinality as $|C_j|$. From the compact support of the Fourier transform of the Meyer wavelet, we have

$$C_j \subset \left(\frac{2\pi}{3}\right)[-2^{j+2}, -2^j] \cup [2^j, 2^{j+2}].$$

With suitably chosen j_0, j_1, and $\eta > 0$, the main steps of the ordinary WaveD are as follows.

1070-9908/$20.00 © 2006 IEEE

1) Compute Fourier coefficients Y_l, G_l, and compute $\hat{\beta}_\kappa = \sum_l (Y_l/G_l)\bar{\Psi}_l^\kappa$ by using Kolaczyck's algorithm, where $\bar{\Psi}_l^\kappa$ is the complex conjugate of Ψ_l^κ.
2) Set the thresholds

$$\lambda_j = \eta\sigma\tau_j\sqrt{\frac{log(n)}{n}}$$

where

$$\tau_j = \left(|C_j|^{-1}\sum_{l\in C_j}|G_l|^{-2}\right)^{\frac{1}{2}}.$$

3) Apply level-dependent thresholding on $\hat{\beta}_\kappa$. Finally, apply the inverse discrete Meyer wavelet transform on the thresholded wavelet coefficients estimate to obtain $\tilde{f}$, an estimate of f.

Essentially, the above thresholding algorithm is based on the following idealized independent normal model:

$$\tilde{\beta}_\kappa = \beta_\kappa + \sigma n^{-\frac{1}{2}}2\sqrt{\pi}\tau_j z_\kappa \quad (3)$$

where z_κ are white Gaussian noise. Then the level-dependent universal threshold [1] is

$$\lambda_j^{UV} = \eta\sigma\tau_j\sqrt{\frac{log(n)}{n}} \quad (4)$$

for $\eta = \sqrt{8\pi}$.

Note that $\sigma^2 4\pi\tau_j^2/n$ is an upper bound of the actual

$$var(\hat{\beta}_\kappa) = \sigma^2 n^{-1}2^{-j}\sum_l\left|\frac{\hat{\psi}(2^{-j}\cdot 2\pi l)}{G_l}\right|^2$$

for the argument of the asymptotic theory in [4, Eq. (48)]. The

$$\sigma_j = \left(2^{-j}\sum_l\left|\frac{\hat{\psi}(2^{-j}\cdot 2\pi l)}{G_l}\right|^2\right)^{\frac{1}{2}} \quad (5)$$

is usually smaller than $2\sqrt{\pi}\tau_j$.

The exact model for unknown parameters β_κ is

$$\hat{\beta}_\kappa \sim N\left(\beta_\kappa, \sigma^2 n^{-1}\sigma_j^2\right).$$

The $\hat{\beta}_\kappa$'s are correlated with

$$cov\left(\hat{\beta}_{\kappa_1}, \hat{\beta}_{\kappa_2}\right) = \sigma^2 n^{-1}\sum_l\frac{\bar{\Psi}_l^{\kappa_1}\Psi_l^{\kappa_2}}{G_l^2}.$$

The WaveD does not utilize any covariance information. An attempt to use these covariance information necessitates an iterative deconvolution procedure that is computationally intensive. The WaveD is a noniterative technique.

The algorithm for translation-invariant WaveD fully exploits the periodicity of f and Ψ_κ. It computes the translation-invariant estimate of f over all circulant shifts with complexity only $O(n(log(n))^2)$ without actually going through n individual "cycle-spin" steps. The brute-force "cycle-spin" would apparently cost $O((nlog(n))^2)$. This fast algorithm of translation-invariant WaveD is very attractive. The main steps of the translation-invariant WaveD are as follows.

1) Compute Fourier coefficients Y_l, G_l, and perform deconvolution

$$\tilde{F}_l = \frac{Y_l}{G_l}$$

where $\tilde{F}_l$ is the estimated Fourier coefficients of f.
2) Set the thresholds λ_j.
3) Loop resolution level j from j_0 to j_1 to compute $\tilde{f}_j^H$, the detail estimate of f at resolution level j. The $\tilde{f}_j^H$ is computed by a sequence of operations, including convolution, inverse Fourier transform, thresholding, and Fourier transform. See [2, Sec. 6] for details.

The performance of WaveD depends on the tuning parameters j_0, j_1, and λ_j's. The coarse scale j_0 has the default value 3 and is not as influential as the other tuning parameters. For j_1, the asymptotic theory suggests that

$$2^{j_1} \sim \left(\frac{n}{log(n)}\right)^{\frac{1}{(1+2\nu)}}.$$

For example, when $\nu = 1$, $n = 2048$, we have $j_1 = 3$ by the above guidance. Apparently, this cannot be used for a finite sample. In the WaveD software, the j_1 is set to be the level proceeding $j(100\%)$, where $j(100\%)$ is the smallest level where 100% of thresholding occurs. This leaves the choice of λ_j's more important.

In the direct data case $(y_i = f(t_i) + \sigma n^{-1/2}z_i)$, there is no need to choose j_1 since it is always set to be $J-1$, where $J = log_2(n)$. That is, all the empirical wavelet coefficients are used in the thresholding process for the denoising problem. This is because the variance of wavelet coefficients is considered to be constant across level j in the direct data case, so that a few coefficients that bear significant signal information are distinguishable from the rest that are pure noise. Note that the true wavelet coefficients β_κ of a wide class such as Besov space signals decay exponentially fast with increasing resolution level j. With indirect data as in model (1), it often occurs that the variance of empirical wavelet coefficients $\tilde{\beta}_\kappa$ increases with j, as seen in (5), so that signal can hardly be separated from noise in those fine levels. Consequently, all the wavelet coefficients in those fine levels have to be discarded.

The λ_j's depend on η. Asymptotic theory suggests that η should be large, but for a finite sample size, smaller η may be desirable. The default value in WaveD is set to be $\sqrt{2}$. In the simulation study for Boxcar convolution in [2], η was set to 0.35.

The difficulty facing the choice of j_1 and η deems necessary a data-adaptive approach. We propose a CV-based approach in the next section.

III. Cross-Validation for WaveD

The aim of deconvolution is the minimization of the mean integrated square error (MISE)

$$\text{MISE} = E\|\hat{f} - f\|_2^2.$$

The $\hat{f}$, hence the MISE, depends on the tuning parameters. In practice, the f is unknown, so a tool that mimics MISE has to be devised. CV is such a tool widely used to choose a tuning parameter in many statistical settings. The classic leave-one-out CV is performed by systematically deleting one observation from the construction of an estimate and then comparing the observed value to the predicted value at the deleted point. This simple leave-one-out procedure cannot be directly applied to wavelet deconvolution because the WaveD algorithm [2], [4] only operates on data sets of size that is a power of 2.

In the direct data case, [6] introduced two-fold CV to choose a threshold for wavelet shrinkage estimate. The basic idea is to remove all the odd-indexed observations first, then use the remaining even-indexed observations to get the wavelet estimates of the even-indexed function values, and then compare the odd-indexed observations to the predicted values at odd-indexed points by linearly interpolating the even-indexed function estimates. The same procedure is done for all the even-indexed observations. The CV score function $\mathrm{CVS}(\cdot)$ compares the interpolated wavelet estimates with the left-out observations to form an estimate of a prediction error at a particular threshold. The $\mathrm{CVS}(\cdot)$ is then numerically minimized over values of the threshold. This two-fold CV can be extended to the deconvolution setting.

The detailed steps forming the $\mathrm{CVS}(\eta, j_1)$ is discussed below.

For the given $s = (\eta, j_1)$, from the given data $d_i = (t_i, g_i, y_i)$, $i = 1, \ldots, n$, where $n = 2^J$ for an integer J, remove all the odd-indexed d_i's from the set. This leaves $n/2$ evenly indexed d_i, which are re-indexed from $m = 1, \ldots, n/2$. A function estimate $\tilde{f}_s^E$ is then constructed from the re-indexed d_m by WaveD. The linear interpolation is used to predict the f_{2i-1} by

$$\bar{f}_{s,m}^E = \frac{1}{2}\left(\tilde{f}_{s,m}^E + \tilde{f}_{s,m+1}^E\right), \quad m = 1, \ldots, \frac{n}{2}$$

setting $\tilde{f}_{s,n/2+1}^E = \tilde{f}_{s,1}^E$ because f is assumed to be periodic. Then the left-out odd-indexed observation is predicted by

$$\hat{y}_{s,m}^E = \bar{f}_s^E \star g(t_{2m-1}), \quad m = 1, \ldots, \frac{n}{2}.$$

The $\tilde{f}_s^O$ is computed for the odd-indexed points by WaveD and the interpolant $\bar{f}_s^O$, the predicted even-indexed observation $\hat{y}_s^O$ computed as above. The CV score function compares the predicted with the observed values

$$\mathrm{CVS}(\eta, j_1) = \sum_{m=1}^{\frac{n}{2}} \left\{ \left(y_{2m-1} - \hat{y}_{s,m}^E\right)^2 + \left(y_{2m} - \hat{y}_{s,m}^O\right)^2 \right\}.$$

The computational complexity for getting $\mathrm{CVS}(\eta, j_1)$ is still $O(n(log(n))^2)$. The Matlab code forming the $\mathrm{CVS}(\eta, j_1)$ is given in the Appendix.

We seek the η, j_1 that minimize the $\mathrm{CVS}(\eta, j_1)$, that is

$$\left(\eta^{CV}, j_1^{CV}\right) = \mathrm{argmin}_{\eta>0, j_0<j_1<J} \mathrm{CVS}(\eta, j_1).$$

The simplest way to solve this optimization problem is to solve the minimization problem for each possible value of j_1 first. For $j_1 = j_0 + 1$ to $J - 1$, solve

$$\eta_{j_1}^{CV} = argmin_{\eta>0} \mathrm{CVS}(\eta, j_1)$$

and then select

$$j_1^{CV} = \mathrm{argmin}_{j_0<j_1<J} \mathrm{CVS}\left(\eta_{j_1}^{CV}, j_1\right)$$

and set $\eta^{CV} = \eta_{j_1^{CV}}^{CV}$.

Empirical experiments show that this j_1^{CV} sometimes overestimates the j_1. So we choose the final j_1 as the smaller value of this j_1^{CV} and j_1^{Default}. With $\eta = \eta^{CV}$, the default choice for j_1, j_1^{Default} is determined from the data as follows. It is set to be the level proceeding the smallest level, where 100% of thresholding occurs in ordinary WaveD [4].

IV. Simulation Results

We present some simulation results to compare the WaveD with fixed η and j_1, with fixed η but default j_1 and the CV tuning based on four artificial signals borrowed from the statistical wavelet literature [2]. Each of the four test signals, (a)Lidar, (b)Bumps, (c)HeaviSine, and (d)Doppler exhibits some inhomogeneous behavior.

With fixed η, the default choice for j_1 is determined from the data by setting it as the level proceeding the smallest level where 100% of thresholding occurs in ordinary WaveD [4].

The results presented in Table I are based on 1000 independent simulations with $n = 2048$ for Gamma blur with medium noise level $\sigma = 0.5$. The corresponding blurred signal-to-noise ratio (BSNR) (BSNR $= 20log_{10}(\|f \star g - \mu(f \star g)\|_2/\sigma)$, where $\mu(f \star g)$ denotes the mean of the blurred signal $f \star g$ samples) is 36.70, 33.18, 48.55, and 27.88, respectively, for Lidar, Bumps, HeaviSine, and Doppler signals. For Gamma blur, $g(t)$ is the probability density function of the $\Gamma(1, 0.0065)$ distribution. Such filter with DIP $= 1$ is often referred to as smooth convolution in the statistical literature since its Fourier coefficients G_l decay homogeneously [4].

The results presented in Table II are based on 1000 independent simulations with $n = 2048$ for Boxcar blur with medium noise level $\sigma = 0.5$. The corresponding BSNR is 36.27, 30.39, 48.41, and 26.92, respectively, for Lidar, Bumps, HeaviSine, and Doppler signals. For Boxcar blur, $g(t) = (1/2a)I_{[-a,a]}(t)$ with $a = 1/\sqrt{353}$, DIP $= 1$.

In both tables, the best performance is obtained with fixed tuning parameters in the first row as used in the simulation study of [2]. The soft thresholding rule gave poorer results than the hard thresholding, whether using a fixed tuning parameter or a CV-based tuning parameter. The default value $\eta = \sqrt{2}$, $j_1 = j_1^{\mathrm{Default}}$ perform worst. It tends to select the j_1 larger than the one in the first row. CV tuning is close to the best performance. Similar patterns were observed when using small $\sigma = 0.01$ and large $\sigma = 1$ and also when using $n = 1024$.

The best performance obtained in the Table I and II with fixed η and j_1 is not practically obtainable because it is not clear how the η and j_1 are chosen. The CV tuning provides an operational tool that mimics the best.

V. Conclusion

This letter has introduced twofold cross-validation to the wavelet deconvolution in a periodic setting. Simulation results show that this twofold cross-validation is adept at selecting a threshold and the finest resolution level. The cross-validation method extends to image deconvolution in a straightforward manner.

TABLE I
MONTE CARLO APPROXIMATIONS TO RMISE = $\sqrt{E\|\hat{f} - f\|_2^2}$.
SMOOTH BLUR DIP = 1

Tuning	Thresh	Lidar	Bumps	HeaviSine	Doppler
$j_1 = 6,\ \eta = \sqrt{2}$	Hard	0.1374	0.2983	0.1277	0.0920
$j_1 = 6,\ \eta = \sqrt{2}$	Soft	0.1844	0.3807	0.1603	0.1126
$j_1 = 5,\ \eta = \sqrt{6}$	Hard	0.1526	0.4255	0.1389	0.0982
j_1 default, $\eta = \sqrt{2}$	Hard	0.2158	1.0621	0.2052	0.1320
CV	Hard	0.1417	0.3039	0.1349	0.1035

TABLE II
MONTE CARLO APPROXIMATIONS TO RMISE = $\sqrt{E\|\hat{f} - f\|_2^2}$.
BOXCAR BLUR DIP = 1.5

Tuning	Thresh	Lidar	Bumps	HeaviSine	Doppler
$j_1 = 4,\ \eta = 0.35$	Hard	0.2435	0.5020	0.1917	0.1313
$j_1 = 4,\ \eta = 0.35$	Soft	0.2705	0.5240	0.2149	0.1395
$j_1 = 4,\ \eta = \sqrt{6}$	Hard	0.3007	0.5417	0.2295	0.1636
j_1 default, $\eta = \sqrt{2}$	Hard	0.4215	0.9521	0.3834	0.3377
CV	Hard	0.2729	0.5308	0.2562	0.1823

In [6], a leave-one-out cross-validation algorithm has also been devised for the denoising problem, which works for data sets with any sample size. It is an interesting research topic to see if this more computationally intensive cross-validation can bring benefits in the wavelet deconvolution context.

APPENDIX
MATLAB SOURCE CODE FORMING THE TWO-FOLD CROSS-VALIDATION SCORE FUNCTION

```
function y = CVS(eta, yobs, g, L, deg, F);
%Two fold cross-validation score function
%Inputs (required):
% eta: the smoothing parameter
% yobs = f * g + Noise
% g: Sample of the (known) function g
% Inputs (optional):
% L: Lowest resolution level (default = 3)
% deg deg of the Meyer Wavelet (default = 3)
% F Finest resolution level (default = J - 1)
% Outputs
% y=Two fold cross-validation score
% function
n = length(yobs);
J = log2(n);

if nargin < 6, F = J - 1; end
if nargin < 5, deg = 3; end
if nargin < 4, L = 3; end
index_even = 2 * (1 : 1 : n/2);
yobs_even = yobs(index_even);
g_even = g(index_even);

index_odd = index_even - 1;
yobs_odd = yobs(index_odd);
g_odd = g(index_odd);

f_estimate_even = TIwaveD1(yobs_even,g_even,L,
deg,F,eta);
f_predict_odd = interp1(index_even, f_estimate_
even,...
index_odd(2:end), 'linear');
f_predict_odd = [0.5 * (f_estimate_even(end) + ...
f_estimate_even(1)), f_predict_odd];
ypredict_odd = real(ifft(fft(g_odd). * ...
fft(f_predict_odd)));

f_estimate_odd = TIwaveD1(yobs_odd,g_odd,L,
deg,F,eta);
f_predict_even = interp1(index_odd, f_estimate_
odd,...
index_even(1:(end-1)), 'linear');
f_predict_even = [f_predict_even, 0.5 * ...
(f_estimate_odd(end)+f_estimate_odd(1))];
ypredict_even = real(ifft(fft(g_even).*...
fft(f_predict_even)));

y = errorLp(yobs_odd, ypredict_odd, 2) + ...
errorLp(yobs_even,ypredict_even,2);
```


ACKNOWLEDGMENT

The authors would like to thank the associate editor and two anonymous reviewers for their constructive remarks.

REFERENCES

[1] D. L. Donoho and I. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, pp. 425–455, 1994.

[2] D. L. Donoho and M. Raimondo, "Translation invariant deconvolution in a periodic setting," *Int. J. Wavelets, Multires., Inf. Process.*, vol. 14, no. 4, pp. 415–432, 2004.

[3] ——, "A fast wavelet algorithm for image deblurring," *Australian New Zealand Ind. Appl. Math. J.*, vol. 46, pp. C29–C46, 2005.

[4] I. M. Johnstone, G. Kerkyacharian, D. Picard, and M. Raimondo, "Wavelet deconvolution in a periodic setting," *J. R. Stat. Soc.*, ser. B, vol. 66, no. 3, pp. 547–573, 2004.

[5] E. Kolaczyk, "Wavelet methods for the inversion of certain homogeneous linear operators in the presence of noisy data," Ph.D. dissertation, Dept. Stat., Stanford Univ., Stanford, CA, 1994.

[6] G. P. Nason, "Wavelet shrinkage using cross-validation," *J. R. Stat. Soc.*, ser. B, vol. 58, pp. 463–479, 1996.